|  |  |
| --- | --- |
| **JEPPIAAR NAGAR, RAJIVGANDHI SALAI**  **CHENNAI – 600119.**  **DEPARTMENT OF INFORMATION TECHNOLOGY**  **IV YEAR B.TECH – VII SEM**  **ACADEMIC YEAR 2024 - 25 (ODD SEM)**  **NM1042 - MERN Stack Powered by MongoDB**  **Online Learning Platform Project Report**  **(Naan Mudhalvan Project)** | |
| **TEAM MEMBERS**  Maha S  Harivarshinee JP  Mohana Priya K  Esmath E  Divya Rakshida R | - 310821205043  - 310821205029  - 310821205048  - 310821205022  - 310821205020 |

|  |
| --- |
| **JEPPIAAR NAGAR, RAJIVGANDHI SALAI, CHENNAI – 600119.**  **DEPARTMENT OF INFORMATION TECHNOLOGY**  This is a Bonafide Record Work of  Register No. submitted for the Anna University Practical  Examination held on in **NM1042 - MERN Stack Powered by**  **MongoDB** during the year .  **Signature of the Faculty-In-Charge Signature of the HOD**  **Date: Examiners Internal:**  **External:** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **COLLEGE VISION & MISSION**  **Vision**  To build Jeppiaar Engineering College as an institution of academic excellence in technological and management education to become a world class university.  **Mission**  • To excel in teaching and learning, research and innovation by promoting the principles of scientific analysis and creative thinking.  • To participate in the production, development and dissemination of knowledge and interact with national and international communities.  • To equip students with values, ethics and life skills needed to enrich their lives and enable them to contribute for the progress of society.  • To prepare students for higher studies and lifelong learning, enrich them with the practical skills necessary to excel as future professionals and entrepreneurs for the benefit of Nation’s economy.  **Program Outcomes**   |  |  | | --- | --- | | **PO1** | **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems. | | **PO2** | **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences. | | **PO3** | **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations. | | **PO4** | **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions. | | **PO5** | **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations. | | **PO6** | **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent  responsibilities relevant to the  professional engineering practice. | | **PO7** | **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development. | | **PO8** | **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice. | | **PO9** | **Individual and teamwork**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings. | | **PO10** | **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions. | | **PO11** | **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments. | | **PO12** | **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change. | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **DEPARTMENT OF INFORMATION TECHNOLOGY**  **Vision**  To produce engineers with excellent knowledge in the field of Information Technology through scientific and practical education to succeed in an increasingly complex world.  **Mission**  • To demonstrate technical and operational excellence through creative and critical thinking for the effective use of emerging technologies.  • To involve in a constructive, team-oriented environment and transfer knowledge to enable global interaction.  • To enrich students with professional integrity and ethical standards that will make them deal social challenges successfully in their life.  • To devise students for higher studies and perpetual learning, upgrade them as competent engineers and entrepreneurs for country’s development.  **Program Educational Objectives (PEOs)**   |  |  | | --- | --- | | PEO 1 | To support students with substantial knowledge for developing and resolving mathematical, scientific and engineering problems | | PEO 2 | To provide students with adequate training and opportunities to work as a collaborator with informative and administrative qualities | | PEO 3 | To shape students with principled values to follow the code of ethics in social and professional life | | PEO 4 | To motivate students for extensive learning to prepare them for graduate studies, R&D and competitive exams | | PEO 5 | To cater the students with industrial exposure in an endeavor to succeed in the emerging cutting-edge technologies |   **Program Specific Outcomes**   |  |  | | --- | --- | | **PSO1** | Students are able to analyze, design, implement and test any software with the programming and testing skills they have acquired. | | **PSO2** | Students are able to design algorithms, data management to meet desired needs, for realtime problems through analytical, logical and problem solving skills. | | **PSO3** | Students are able to provide security solutions for network components and data storage & management which will enable them to work in the industry ethically. |   **Course Outcomes (COs)**   |  |  | | --- | --- | | **C407.1** | Configure various virtualization tools such as Virtual Box, VMware workstation | | **C407.2** | Design and deploy a web application in a PaaS environment | | **C407.3** | Learn how to simulate a cloud environment to implement new schedulers. | | **C407.4** | Install and use a generic cloud environment that can be used as a private cloud. | | **C407.5** | Install and use Hadoop | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Table of Contents**   |  |  | | --- | --- | | **S.No** | **Topic** | | 1 | Introduction | | 2 | Project Overview | | 3 | Architecture | | 4 | Setup Instructions | | 5 | Folder Structure | | 6 | Running the Application | | 7 | API Documentation | | 8 | Authentication | | 9 | User Interface | | 10 | Testing | | 11 | Screenshots or Demo | | 12 | Known Issues | | 13 | Future Enhancements | |

|  |
| --- |
| 1. **Introduction**   **Project Title**: Online Learning Platform (MERN Stack)  This project provides an online platform where students and teachers can interact and engage in a seamless learning experience. Students can browse a wide range of courses, enroll in them, and participate in live sessions, while teachers have the ability to create, manage, and update their courses with ease. The platform also facilitates secure payment processing, ensuring that students can access their courses without issues. In addition, it offers a robust discussion forum, fostering communication and collaboration between students and instructors. The goal of the project is to build an intuitive, secure, and efficient online learning environment that guarantees a smooth, enjoyable experience for both students and teachers, while supporting ongoing growth and development in the learning process.  **Team Members and Roles**:   1. Maha S [310821205043]: Team lead and Unit tester- As the Team Lead, coordinated to technical guidance. Unit Testing was conducted to identify and resolve bugs, ensuring the stability and functionality of all features across the platform. 2. Harivarshinee Jp [310821205029]:Quality Assurance and API Testing- Performed comprehensive Quality Assurance to ensure the platform’s functionality, stability, and performance. Conducted API testing to validate endpoints, verify responses, and ensure seamless integration between the frontend and backend. 3. Mohana Priya K [310821205048]: Backend Developer - Responsible for API development, database integration, and deployment. Designed and implemented server-side logic to ensure efficient data handling and smooth communication between the frontend and backend. 4. Esmath E [310821205022]: Frontend Developer - Designed the user interface, implemented React components, and ensured responsive design. Collaborated with the team to create an intuitive and seamless user experience across different devices. 5. Divya Rakshida R [310821205020]: Database Manager - Designed and managed MongoDB schemas, ensured data consistency, and optimized queries. Monitored database performance and implemented improvements to enhance scalability and reliability.   **2. Project Overview**  **Purpose**:  The Online Learning Platform (OLP) is a web application designed to provide students and instructors with an easy-to-use, efficient way to access and manage educational content. The platform solves the problem of students struggling to find good-quality courses and instructors needing a simple tool to manage their courses. This platform makes it easy for students to enroll  in courses and track their learning, while instructors can easily create and manage courses.  The goal of the platform is to create a smooth, user-friendly experience that helps both students and teachers interact easily. It integrates features like course management, real-time communication, and progress tracking, all in one place, ensuring a seamless learning process**.**  **Key Objectives**:  1.Easier Access to Learning: Provide students with the ability to access courses anytime,anywhere.  2.Easy Course Management: Allow instructors to create and organize courses efficiently.  3.Better Interaction: Promote communication between students and instructors with discussion forums and live sessions.  4.Safe Payment Options: Provide secure payment methods for paid courses.  5.Responsive and Scalable: Ensure the platform works well on all devices and can handle a growing number of users.  **Features**:  1. User Authentication  The platform uses secure authentication methods to ensure that only authorized users, such as students and instructors, can log in and access their accounts. Registration and login require email verification to protect user information.  Passwords are securely encrypted before they are stored in the database, making sure that user credentials are protected from unauthorized access. This encryption method helps safeguard user data, even in the event of a security breach.  2. Profile Management  Students can create profiles that include personal details, learning progress, enrolled courses, and feedback on completed courses. They can update their preferences, including course suggestions and notifications, ensuring they receive personalized learning experiences.  Instructors can create and update their profiles, showcasing their expertise, qualifications, and courses they are offering. This allows students to evaluate instructors based on their professional background and teaching materials.  3. Course Management & Enrollment  Instructors can create courses, upload materials (such as videos, documents, and quizzes), and update course content as needed. They can track student progress and interact with students through discussion forums or direct messaging.  Students can browse the available courses, read course descriptions, view ratings, and enroll in courses that align with their learning goals. They can access course content at their own pace and review materials as needed.  Admins can manage course listings, ensuring that the course catalog is up to date. They can also approve new courses and remove any outdated or inappropriate content.  4. Real-Time Communication  The platform allows students and instructors to communicate instantly through live chat features or messaging within the platform. Whether it’s for course discussions, answering questions, or providing additional guidance, communication is direct and efficient.  Discussion Boards: Each course has a forum for students to engage with instructors and peers, ask questions, and share resources.  Notifications: Users will receive notifications for new messages, course updates, or announcements, ensuring they never miss important information.  5. Payment Processing  Secure Payment Options: Students can pay for courses using a variety of secure payment methods. The platform supports one-time payments for individual courses as well as subscription models for access to premium content.  The payment system ensures that transactions are processed safely and efficiently, and invoices are generated automatically after a payment is made.  Escrow Services: For paid courses or premium features, payments will be processed through secure gateways. For subscription-based services, students can cancel or renew their access easily within the platform.  6. Admin Panel  The Admin Panel provides administrators with full control over the platform. Admins can monitor user activity, manage course content, and resolve any issues that arise within the platform.  User Management: Admins can manage both student and instructor accounts, approve or deactivate accounts, and handle any disputes.  Analytics and Reports: Admins can access detailed reports to track user engagement, revenue, and course performance. This helps in making informed decisions about platform improvements.  7. Search and Filters  The platform includes an advanced search function to help students easily find courses by title, subject, instructor, or price.  Filters are available to narrow down course choices based on the level (beginner, intermediate, advanced), course type (video-based, reading material, assignments), and availability (online, live).  Personalized Recommendations: Based on their learning history and preferences, students will receive suggestions for courses they might be interested in, making the learning experience more tailored to their needs. Architecture **Frontend Architecture**  The frontend of the Online Learning Platform is built with React.js, leveraging a component-based structure to ensure modularity, reusability, and maintainability. The architecture is designed to be scalable, supporting both small-scale courses and large, complex learning environments. Key aspects of the frontend architecture include:  **- Component-Based Design:**  Each major feature of the platform, such as course listings, user profiles, video players, and discussion forums, is developed as a separate React component. This modular approach allows for easy maintenance and ensures that components can be reused and extended as the platform evolves. Components are independent, making it simple to update or add new features without disrupting the overall system.  **- State Management:**  The platform uses Redux Toolkit for global state management. Redux provides a centralized store to manage application state, enabling real-time synchronization across components. Whether users are interacting with course content, submitting assignments, or tracking progress, Redux ensures that updates are consistently reflected across all views in the platform. This helps maintain smooth and seamless user interactions, especially when managing dynamic data such as course progress, notifications, and chat messages.  **- Routing:**  **React Router** is utilized to handle navigation between different pages such as the Home, Course Catalog, Profile, and Discussion forums. This allows users to transition smoothly between pages without requiring full page reloads. Additionally, **React Router Hash Link** is used to provide smooth scrolling for internal anchor links, enhancing the user experience by navigating directly to specific sections within a page (e.g., lesson content or quizzes)  **- UI/UX Design:**  The platform’s design is crafted using **CSS**, focusing on a responsive, intuitive user experience. A mobile-first approach ensures the platform is fully optimized for mobile devices, offering a seamless experience for learners on the go. Once the mobile version is perfected, the design is progressively enhanced for larger screens, offering an intuitive, user-friendly interface for all users. This ensures accessibility and ease of use, whether on smartphones, tablets, or desktop computers.  **Backend Architecture**  The backend of the Online Learning Platform is built using **Node.js** and **Express.js**, which together form a robust RESTful API. The backend handles critical operations such as user authentication, course management, content delivery, and real-time communication. It provides the foundation for user interactions, including course enrollment, progress tracking, and forum discussions. Key aspects of the backend architecture include:  **- API Routes:**  The backend exposes a range of API routes that manage interactions with the platform's data, including user accounts, course content, quizzes, and discussions. These routes handle requests for creating, updating, and retrieving courses, assignments, student progress, and feedback. The API ensures secure communication and enables seamless integration between the frontend and backend, supporting all key features like course enrollment, lesson completion tracking, and certificate issuance.  **- User Authentication & Authorization:**  The platform uses secure authentication methods to ensure that only authorized users, such as students and instructors, can log in and access their accounts. Registration and login require email verification to protect user information.  Passwords are securely encrypted before they are stored in the database, making sure that user credentials are protected from unauthorized access. This encryption method helps safeguard user data, even in the event of a security breach.  **- Real-Time Communication:**  Real-time messaging and notifications are powered by **WebSockets** (or Socket.IO), enabling features like live chat between instructors and students, as well as instant notifications for new course updates, assignments, and announcements. This ensures that users receive up-to-date information and can engage with each other in real-time, enhancing the learning experience.  **- Database Integration:**  The platform uses **MongoDB** (or another NoSQL database) for storing dynamic content such as courses, user data, forum posts, and quiz results. MongoDB provides flexibility and scalability, allowing for rapid growth and easy management of large amounts of unstructured data. The database is connected through an **ORM/ODM** like **Mongoose**, which simplifies data modeling and querying.  - **Course Management & Content Delivery:**  The backend handles the management of courses, lessons, video streaming, and assignments. Instructors can create and update course materials, and students can access lessons based on their enrollment status. Content delivery is optimized for performance, ensuring fast loading times for video lessons and smooth navigation across courses and quizzes.  **Database Architecture**  The database for the Online Learning Platform is organized into collections, each dedicated to specific aspects of the platform. These collections store and manage the core data necessary for delivering an engaging and efficient learning experience. The key collections include:  **Users:**  The Users collection stores all user-related data, including authentication details, contact information, and profiles. It distinguishes between different roles such as students, instructors, and administrators.  Students: This includes information such as enrolled courses, progress, completed assignments, and exam scores.  Instructors: This contains details such as courses they teach, course materials, ratings, and student feedback.    The collection also includes profile settings, course progress, certifications, and preferences. Instructors can update their teaching materials, while students can track their learning journey.  **Courses:**  The Courses collection holds detailed information about all available courses on the platform. Each document contains:  Course description: Overview, learning objectives, and course content.  Instructor details: The instructor(s) responsible for the course.    Course content: Including video lessons, articles, quizzes, and assignments.  Enrollment information: Which students are enrolled, and their progress in the course.  Ratings and reviews: Feedback from students who have completed the course. This collection helps manage course creation, content updates, and enrollment details.  **Payments:**  The Payments collection stores transaction details for premium courses or subscriptions. It includes:  Transaction ID: Unique identifier for each transaction.  Payment amount and status: Details of payments for course enrollments, subscriptions, or other services.  User details: Information about the student who made the payment. This collection facilitates tracking payments for premium content and subscriptions.   1. **Setup Instructions**   **Prerequisites**  Before setting up the project locally, make sure you have the following software  installed on your machine:  - Node.js: Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. It  is required to run both the frontend and backend of this project. You can  download Node.js from the official website [here](https://nodejs.org/).  - MongoDB: MongoDB is a NoSQL database used to store user data, job postings,  bids, and messages. You can either install MongoDB locally or use MongoDB Atlas,  a cloud-based solution, for easier setup and management.  - npm (Node Package Manager): npm is used to install the necessary  dependencies for both the frontend and backend. npm comes pre-installed with  Node.js, so you will have it once you install Node.js.  **Installation Steps:**  **Clone the Repository**  Begin by cloning the project repository to your local machine using the following command:  git clone <repository-url>  This will download the project files to your local machine.  **Install Dependencies**  After cloning the repository, navigate to the following directories to install the necessary npm packages:  **For the Frontend:** Navigate to the frontend directory, which contains all the frontend code built with React. Run the following command to install the required dependencies:  cd frontend/  npm install  This will install all the packages listed in the frontend/package.json file, including React and related libraries.  **For the Backend:** Similarly, navigate to the backend directory, which contains the backend code built with Node.js and Express. Run the following command:  cd backend/  npm install  This will install all the necessary backend dependencies such as Express, Mongoose, jsonwebtoken, bcryptjs, dotenv, and others required for API functionality.  **Setup Environment Variables**  To ensure the frontend, backend, and socket server communicate securely, you need to set up the environment variables:  Create .env Files: Create a .env file in the root directory for both the frontend and backend.  **Backend (backend/.env):** Include the following environment variables in the .env file located in the backend directory:  env  MONGODB\_URI=<your-mongodb-uri>  PORT=5000  MONGODB\_URI: The MongoDB connection string. If using MongoDB Atlas, you’ll find this in your dashboard. For local setups, it could look like mongodb://localhost:27017/your-database-name.  **For the Backend (API):** In one terminal window, navigate to the backend directory and start the backend server:  cd backend/  npm start  The backend will now be running at [http://localhost:5000](http://localhost:3001/" \t "https://chatgpt.com/c/_new).  For the Frontend: In another terminal window, navigate to the frontend directory and start the frontend server:  cd frontend/  npm start  The frontend will be available at [http://localhost:5](http://localhost:3000/" \t "https://chatgpt.com/c/_new)173.  **For the Socket Server (if applicable):** If your project includes a socket server for real-time communication, navigate to the socket directory (if available) and start the socket server:  cd socket/  npm start  The socket server will provide real-time capabilities between the frontend and backend, facilitating messaging and live updates.  **Access the Application**  Once all the servers are running, you can access the application in your web browser:  Frontend: Open your browser and go to [http://localhost:5173](http://localhost:3000/" \t "https://chatgpt.com/c/_new) to interact with the user interface of the application.  Backend API: The backend API can be accessed at [http://localhost:5](http://localhost:3001/" \t "https://chatgpt.com/c/_new)000. The frontend communicates with this API to fetch data and perform actions.  Socket Server: If your application uses real-time communication, the socket server is responsible for facilitating messaging between the frontend and backend. You can test real-time features by interacting with the frontend and observing the communication between users.   1. **Folder Structure**   **Frontend Folder**  The frontend folder contains all the files for the client-side application built with React.js.  frontend  ├── node\_modules/ # Dependencies installed for the frontend.  ├── public/ # Contains public files like `index.html` and `favicon.ico`.  └── index.html # Main HTML file rendered by React.  └── src/ # Source code for the React frontend.  ├── assets/ # Static files like images, fonts, and icons.  └── Images/ # Folder for images used in the frontend.  └── bg.jpg # Background image used on the homepage.  ├── components/ # Reusable React components.  ├── admin/ # Admin section components.  ├── AdminHome.jsx  ├── AllCourses.jsx  │ ├── common/ # Common components used across sections.  │ │ ├── AllCourses.jsx  │ │ ├── AxiosInstance.jsx  │ │ ├── Dashboard.jsx  │ │ ├── Home.jsx  │ │ ├── Login.jsx  │ │ ├── NavBar.jsx  │ │ ├── Register.jsx  │ ├── user/ # User-specific components.  │ │ ├── student/ # Components for students (e.g., course content).  │ │ │ ├── CourseContent.jsx  │ │ │ ├── EnrolledCourses.jsx  │ │ │ ├── StudentHome.jsx  │ │ ├── teacher/ # Teacher components (e.g., add/edit courses).  │ │ │ ├── AddCourse.jsx  │ │ │ ├── TeacherHome.jsx  ├── App.css # Styling for the React app.  ├── App.jsx # Root component managing routing.  ├── main.jsx # Entry point for the React app.  ├── index.html # Main HTML file rendered by React.  └── vite.config.js # Vite configuration file.  **Backend Folder**  The server folder contains the backend-related files, which include the API routes, controllers, models, and other necessary server-side logic.  backend  ├── config/ # Contains configuration files for database and environment variables.  ├── controllers/ # Controllers handling business logic.  │ ├── adminController.js # Admin-specific logic.  │ ├── userController.js # User management logic.  ├── middlewares/ # Middleware functions (e.g., authentication).  │ └── authMiddleware.js # Middleware for securing routes.  ├── models/ # Mongoose models for MongoDB collections.  │ ├── courseModel.js # Course schema.  │ ├── coursePaymentModel.js # Payment schema for courses.  │ ├── enrolledCourseModel.js # Tracks enrolled courses.  │ ├── userModel.js # User schema.  ├── routes/ # Defines Express.js routes for the API.  │ ├── adminRoutes.js # Routes for admin actions.  │ ├── userRoutes.js # Routes for user management.  ├── .gitignore # Git ignore file to exclude unnecessary files.  ├── index.js # Entry point for the backend server.  ├── package-lock.json # Lock file for consistent dependency installations.  └── package.json # Defines backend dependencies and scripts.  **Socket Folder**  The socket folder is responsible for managing real-time communication using Socket.io.  socket/  ├── node\_modules/ # Dependencies for the socket server.  ├── index.js # Main file for initializing the Socket.io server.  └── .env # Environment variables for the socket server.   1. **Running the Application**   To run the Online Learning Platform locally, follow the steps below:  **Frontend:**  Navigate to the frontend directory:  cd frontend/  1.Install the required dependencies:  npm install   1. Start the frontend server:   npm start  The frontend will be available at http://localhost:5173.  **Backend:**  1.Navigate to the backend directory:  cd backend/  2.Install the required dependencies:  npm install  3.Start the backend server:  npm start  The backend will be available at http://localhost:5000.  **Socket Server:**  Navigate to the socket directory:  cd socket/  1.Install the required dependencies:  npm install  2.Start the socket server:  npm start  The socket server will run on the backend, facilitating real-time communication between the frontend and backend.   1. **API Documentation**   The backend exposes several endpoints for managing users, courses, enrollments, and payments. Below is the detailed API documentation for each endpoint:    **User Authentication**  POST /api/auth/register  Description: Register a new user (student or teacher).  Request Body:  {  "name": "student1",  "email": "student1@gmail.com",  "password": "student1",  "role": "student"  }  Response:  {  "message": "User registered successfully"  }  POST /api/auth/login  Description: Login a user and return a JWT token.  Request Body:  {  "email": "student1@gmail.com",  "password": "student"  }  Response:  {  "token": "JWT\_Token\_Here"  }  **Course Management**  POST /api/courses  Description: Create a new course (teacher only).  Request Body:  {  "title": "Introduction to React",  "description": "Learn the basics of React.js",  "price": 200,  "duration": "4 weeks"  }  Response:  {  "message": "Course created successfully"  }  GET /api/courses  Description: Retrieve all available courses.  Response:  [  {  "title": "Introduction to React",  "description": "Learn the basics of React.js",  "price": 200,  "duration": "4 weeks"  },  {  "title": "Mastering Node.js",  "description": "An advanced course on Node.js for backend development.",  "price": 300,  "duration": "6 weeks"  }]  **Enrollment Management**  POST /api/enrollments  Description: Enroll a student in a course.  Request Body:  {  "courseId": "course\_id\_here",  "studentId": "student\_id\_here"  }  Response:  {  "message": "Enrollment successful"  }  GET /api/enrollments/{studentId}  Description: Retrieve all courses a student is enrolled in.  Response:  [  {  "courseId": "course\_id\_here",  "title": "Introduction to React",  "status": "In Progress"  },  {  "courseId": "course\_id\_here",  "title": "Mastering Node.js",  "status": "Completed"  }]  **Payment Management**  POST /api/payments  Description: Process a payment for a course.  Request Body:  {  "studentId": "student\_id\_here",  "courseId": "course\_id\_here",  "amount": 200,  "paymentMethod": "credit\_card"  }  Response:  {  "message": "Payment processed successfully"  }  GET /api/payments/{studentId}  Description: Retrieve all payment records for a student.  Response:  [  {  "paymentId": "payment\_id\_here",  "courseTitle": "Introduction to React",  "amount": 200,  "status": "Successful"  },  {  "paymentId": "payment\_id\_here",  "courseTitle": "Mastering Node.js",  "amount": 300,  "status": "Successful"  }]  **8. Authentication**  The authentication system for the Online Learning Platform ensures secure management of user sessions and access to protected resources.  **User Registration**  When a user (student or teacher) registers, the system hashes the password using Bcrypt to ensure secure storage.  The user details, such as name, email, role, and hashed password, are saved in the MongoDB database.  **Login**  During login, the user's credentials are validated against the database.  If the credentials are valid, the user gains access to the platform and can perform authorized actions.  If the credentials are invalid, an appropriate error message is returned to the user.  **Authorization**  Authorization ensures that users can only access features relevant to their role:  Students: Can view available courses, enroll in courses, and process payments.  Teachers: Can create, update, and manage course content.  Admins: Can oversee platform activity, including user and course management.  Unauthorized users attempting to access restricted features are denied access.  **Payment Authentication**  When a student initiates a course payment, the system validates the user’s identity and checks enrollment eligibility.  Payment requests must include the user’s valid session or credentials to prevent unauthorized transactions.  Only authenticated students can complete payment for courses, ensuring secure access to the content.  **Session Management**  The frontend maintains session information to ensure the user remains logged in during their interaction with the platform.  Session information is securely stored and included in all requests for actions such as enrolling in courses, making payments, or accessing course materials.  The session automatically expires after a defined period, requiring the user to log in again for continued access.  **9.User Interface**  The online learning platform is designed with a clean, modern interface focused on enhancing the learning experience for students. Below are some of the key components of the user interface:  - Homepage:  The homepage offers a clear overview of the platform, featuring a prominent search bar to find courses, as well as curated course recommendations. It also provides options for registering and logging in, ensuring a seamless entry point for new and returning users.  - Dashboard:  The dashboard provides a personalized view for each student, allowing them to track their progress, access enrolled courses, and manage their learning activities efficiently. Key sections include a summary of ongoing courses, upcoming assignments, and a personalized learning calendar.  - Course Catalog:  This section displays a comprehensive list of available courses, organized by categories such as programming, design, business, and personal development. Users can filter courses based on their interests, skill level, and learning goals.  - Course Page:  Each course page provides detailed information about the course, including its curriculum, instructor profile, reviews, and pricing. Students can preview course content, enroll in courses, and access course materials.  - Learning Environment:  The learning environment offers a user-friendly interface for accessing course content, such as video lectures, quizzes, assignments, and discussion forums. It provides tools for note-taking, bookmarking, and progress tracking.  **10.Testing**  Testing is essential to ensure the platform performs as expected, providing a seamless learning experience and maintaining security and reliability. Various testing strategies and tools were employed to guarantee the platform’s functionality.  **Unit Testing:**  React Testing Library and Jest were used for testing frontend components. Key UI components like CourseCard, LessonPlayer, and QuizForm were thoroughly tested to ensure they render correctly and behave as expected during user interactions.  Unit tests focused on checking the basic functionality of each component, such as ensuring that form inputs update correctly, video players function properly, and quizzes are graded accurately.  **API Testing:**  Tests were created for key routes, including:  User registration: Ensuring valid input creates a user and invalid input triggers an error.  Course creation: Verifying that courses are correctly added to the database, including their content and metadata.  Lesson playback: Ensuring that video and audio content plays smoothly and that progress is tracked accurately.  Quiz submission: Verifying that quiz answers are graded correctly and that student scores are updated.  **Integration Testing:**  Full-Flow Testing: We tested the integration of frontend and backend to ensure that the user experience flows smoothly from course enrollment to lesson completion and quiz submission.  Mock Data: To simulate real user interaction, mock data was used to test the entire process, including course enrollment, content consumption, and quiz attempts. This ensured that each part of the application works together as expected.  **Performance Testing:**  Tools like Lighthouse were utilized to assess the platform's performance. These tests focused on page load times, responsiveness, and overall efficiency.  Performance testing helped ensure that the platform is optimized for speed, even with large numbers of users and concurrent interactions, and that the UI remains responsive on both desktop and mobile devices.  By utilizing a comprehensive testing strategy, the platform ensures a smooth, reliable, and efficient learning experience for students.   1. **ScreenShots or Demo**     Screenshots:  The screenshots below highlight key features of the online learning platform:  **1. Dashboard:**  The dashboard provides a personalized view for each student, allowing them to track their progress, access enrolled courses, and manage their learning activities efficiently. Key sections include a summary of ongoing courses, upcoming assignments, and a personalized learning calendar.  **WhatsApp Image 2024-11-26 at 17.07.48_4986d535**    dashboard2 |
| **3.Course Catalog:**  This section displays a comprehensive list of available courses, organized by categories such as programming, design, business, and personal development. Users can filter courses based on their interests, skill level, and learning goals.  WhatsApp Image 2024-11-26 at 17.07.47_0f4db2b9  **4.Course Page:**  Each course page provides detailed information about the course, including its curriculum, instructor profile, reviews, and pricing. Students can preview course content, enroll in courses, and access course materials.  Course page 1  **5.Enrolled Courses**  The Enrolled Course feature displays provides a comprehensive overview of the courses you're currently enrolled in. You can easily access course materials, track your progress, and submit assignments. You'll also find a personalized learning calendar that outlines important deadlines and upcoming activities.  Enrolled course 1    **6.Payment for Courses**    This page confirms your successful payment for the specified course or subscription. It provides a detailed breakdown of the transaction, including the total amount paid, payment method, and order number. You can also download a copy of the receipt for your records.  Payment 1      **7.Adding New Course**  This page allows instructors to create new courses and manage their course content. You can upload video lectures, add quizzes, assign projects, and facilitate discussions. You can also set deadlines, track student progress, and provide personalized feedback**.**    **Course addition**  **8.Certification Page**  This page showcases your academic achievements and professional certifications. Here, you can view a detailed list of certifications you've earned, including the issuing organization, date of acquisition, and a brief description of the certification's significance. You can also download and share your certificates with potential employers or academic institutions.        Certificate |
| **12.Known Issues**  Quiz Timing Issues:  In rare cases, quiz timers may not function correctly, leading to unexpected timeouts or premature ending of quizzes. This can be frustrating for students who are in the middle of answering questions. We are thoroughly investigating this issue and working on a fix that will be deployed in an upcoming update. The goal is to ensure that the quiz timer accurately tracks time and prevents premature cutoffs, allowing students to complete their quizzes without issues.  Course Enrollment Issues:  Some users have reported difficulties with enrolling in courses, particularly after making a payment. In certain cases, students are not correctly enrolled, or the course content does not appear immediately after purchase. We are working to resolve these enrollment syncing issues by improving the backend processes that handle course registration and access. Our team is focused on ensuring that once a student enrolls and completes a payment, they gain immediate and reliable access to all course materials.  Payment Processing Delays:  A few students have experienced delays when processing payments for courses. While payments are generally processed smoothly, some transactions may take longer than expected to reflect in the system, causing delays in granting course access. We are investigating this issue and optimizing the payment gateway integration to reduce delays and ensure that students can access their courses promptly after payment.  Discussion Forum Moderation:  We are committed to fostering a positive and respectful learning environment. To ensure this, we are enhancing our moderation tools to promptly address any inappropriate or off-topic discussions. Inappropriate content, such as spam or offensive posts, is actively monitored and removed to maintain a safe environment for all users. Additionally, we are improving our reporting and flagging system so users can easily report issues, and moderators can take swift action.  Course Content Upload Issues:  Some instructors have reported delays or issues when uploading course materials, such as videos, quizzes, and reading materials. In some cases, files may not upload correctly or appear incomplete. We are working to improve the file upload process, optimizing both the backend and frontend to handle larger files and ensure that all content is successfully uploaded and accessible to students without delay.  Email Notification Delays:  There have been occasional delays in sending email notifications for activities such as course enrollment, quiz completion, or assignment feedback. This can lead to students and teachers not receiving timely updates. Our team is investigating the email delivery system to ensure more reliable and faster notifications, and we are optimizing the infrastructure to handle high volumes of email notifications more efficiently. |
| **13.Future Enhancements**  1.Improved Quiz Timing System  To address the current quiz timing issues, we are planning to introduce a more robust and accurate timer system. Future updates will include:  Real-time syncing of quiz timers to prevent time discrepancies across different devices and sessions.  An adaptive timer that adjusts for internet lag or delays, ensuring fair timing for all students.  A countdown feature with visual alerts to notify students when their time is about to expire, enhancing their quiz experience.  2.Enhanced Course Enrollment System  To resolve course enrollment syncing issues, we will implement:  A more reliable backend process for real-time course registration, ensuring that students are immediately enrolled after payment.  Enhanced error handling and user notifications during the enrollment process to ensure users are aware of their status.  A manual override option for admins to quickly resolve any enrollment issues, ensuring students gain access to course materials without delays.  3.Optimized Payment Processing  We are working to improve the speed and reliability of the payment processing system by:  Integrating more payment gateways to provide faster and more secure transaction options.  Expanding real-time transaction tracking for both students and admins, ensuring payments are reflected immediately.  Implementing an automatic refund system in case of failed or delayed payments, ensuring a smoother user experience.  4.Enhanced Discussion Forum Moderation  To improve the discussion forum experience, we plan to introduce:  AI-powered moderation tools to automatically detect and remove inappropriate content.  A more comprehensive reporting system for users to report content directly from the forums, making it easier for moderators to take action quickly.  Improved user controls, allowing users to filter out spam or offensive content more efficiently while still allowing open discussions.  5.Improved Course Content Upload System  To address course content upload issues, we aim to:  Optimize the file upload process to handle larger files more efficiently, including videos, documents, and quizzes.  Introduce drag-and-drop upload features to make content management easier for teachers.  Improve content validation and error checking during uploads to ensure materials are properly formatted and accessible.  6.Faster Email Notification System  We are working on improving the email notification system by:  Implementing a more efficient queuing system to reduce email delivery delays.  Introducing real-time push notifications to supplement emails, ensuring students and teachers receive timely updates about course activities.  Improving the email template design to ensure clarity and better engagement, enhancing user experience across all devices.  **Conclusion**  In conclusion, the development of this online learning platform using the MERN stack has been a successful demonstration of the power and flexibility of modern web technologies. By leveraging React, Node.js, Express.js, and MongoDB, the platform provides a responsive, scalable, and secure environment for learners. The integration of key features such as video lectures, quizzes, discussion forums, and personalized learning paths has created a seamless and engaging learning experience. |
| **Result:**  The Online Learning Platform has been successfully developed using the MERN stack (MongoDB, Express.js, React.js, and Node.js). The platform provides a responsive and intuitive experience for both students and teachers, featuring course enrollment. |